

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In Re Application of	:	LAZARIDIS, <i>et al.</i>
	:	
Application No.	:	10/768,509
	:	
Filed	:	January 30, 2004
	:	
Title	:	System and Method for Implementing a
	:	Natural Language User Interface
	:	
Examiner	:	James S. Wozniak
	:	
TC/A.U.	:	2626
	:	
Docket No.	:	555255-012-690

APPEAL BRIEF

Mail Stop Appeal Brief – Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

This Appeal Brief is filed in response to the final Office Action mailed on December 4, 2008 and pursuant to the Notice of Appeal filed on February 3, 2009 and in further response to the Notice of Panel Decision from Pre-Appeal Brief Review issued on August 26, 2009. Any fees due may be withdrawn from Jones Day Deposit Account No. 501432, ref. 555255-012690.

(I) REAL PARTY INTEREST

The real party in interest is Research In Motion Limited, having its principal place of business at 295 Phillip Street, Waterloo, Ontario, CANADA N2L 3W8, as evidenced by an assignment executed October 11, 2001 and recorded under Reel/Frame 012421/0168.

(II) RELATED APPEALS AND INTERFERENCES

There are no related appeals or interferences to this application.

(III) STATUS OF CLAIMS

Claims 2, 5-13, 37, 41-45, 48-50 and 56 are all pending, finally rejected and currently being appealed. Claims 1, 3, 4, 14-36, 38-40, 46, 47, and 51-55 are cancelled.

(IV) STATUS OF AMENDMENTS

An amendment was filed subsequent to the final rejection to cancel claims 54-55. A subsequent advisory action was issued by the Examiner on February 13, 2009, advising that the amendment would be entered.

(V) SUMMARY OF CLAIMED SUBJECT MATTER

A. Independent Claim 2

Independent claim 2 is directed to a method of launching a software application in a hand-held device. The method includes the step of receiving an abbreviated textual command in a natural language search engine, entered by a user of the hand-held device. An example implementation of this claim element is described in the specification at Figs. 4 and 6, page 5, lines 6-16 and page 6, lines 9-10.

The method further includes the step of while receiving the abbreviated textual command, searching a natural language database that stores a data set of abbreviated textual commands and associated application commands. An example implementation of this claim element is described at Figs. 6 and 7, page 6, line 9- page 7, line 11, and page 9, lines 14-17.

The method further includes the step of while receiving the abbreviated textual command, analyzing historical preferences to determine one or more probable complete commands matching a currently received portion of the abbreviated textual command. An example implementation of this claim element is described at Figs. 6 and 7, page 6, lines 15-17, and page 7, line 14 – page 8, line 8.

The method further includes the step of while receiving the abbreviated textual command, displaying a list of probably complete commands matching the currently received portion of the abbreviated textual command. An example implementation of this claim element is described at Fig. 6, page 8, lines 2-4, page 9, lines 1-5, and page 9, lines 14-17.

The method further includes the step of while receiving the abbreviated textual command, if the user selects a complete command from the list, then setting the complete command as the abbreviated textual command, and executing the complete command. An example implementation of this claim element is described at Fig. 6, page 8, lines 3-4, page 9, lines 11-13 and 16-17.

The method further includes the step of while receiving the abbreviated textual command, if the user does not select a complete command from the list, then receiving an entire abbreviated textual command from the user in the natural language search engine, by the user entering remaining characters of the entire abbreviated command to narrow the list of complete commands. An example implementation of this claim element is describe at Fig. 6, page 8, lines 3-4.

B. Independent Claim 37

Independent claim 37 is directed to a method of receiving abbreviated textual commands. The method includes the step of storing a data set of abbreviated textual commands and corresponding complete commands. An example implementation of this claim element is described at Figs 7, and page 6, line 9- page 7, line 11.

The method further includes the step of receiving a portion of an abbreviated textual command being entered by a user. An example implementation of this claim element is described at Figs. 4 and 6, page 5, lines 6-16 and page 6, lines 9-10.

The method further includes the step of before receiving the entire abbreviated

textual command, comparing the received portion of the abbreviated textual command to the stored abbreviated commands to determine a probably subset of the complete commands. An example implementation of this claim element is described at Figs. 6 and 7, page 6, lines 15-17, and page 7, line 14 – page 8, line 8.

The method further includes the step of displaying the probably subset of the complete commands to the user. An example implementation of this claim element is described at Fig. 6, page 8, lines 2-4, page 9, lines 1-5, and page 9, lines 14-17.

The method further includes the step of, if the user selects one of the complete commands, then executing the selected complete command. An example implementation of this claim element is described at Fig. 6, page 8, lines 3-4, page 9, lines 11-13 and 16-17.

The method further includes the step of if, instead of selected a complete command, the user enters a further portion of the abbreviated textual command, then narrowing the probable subset based on said further portion. An example implementation of this claim element is described at Fig. 6, page 8, lines 3-4.

C. Independent Claim 48

Independent claim 48 is directed to a method of receiving textual commands. The method includes the step receiving a text string being entered by a user. An example implementation of this claim element is described at Figs. 4 and 6, page 5, lines 6-16 and page 6, lines 9-10.

The method further includes the step of while receiving the text string, comparing a

received portion of the text string to stored text commands to determine which of the stored text commands is a probably text command based on a portion of the probable text command matching the received text string. An example implementation of this claim element is described at Figs. 6 and 7, page 6, line 9- page 7, line 11, page 7, line 14 – page 8, line 8, and page 9, lines 14-17.

The method further includes the step of initiating a software operation corresponding to the probable text command. An example implementation of this claim element is described at Fig. 6, page 4, line 17 – page 5, line 1, page 8, lines 3-4, and page 9, lines 11-13 and 16-17.

The method further recites that the comparing and initiating steps are performed without the user having entered a delimiter denoting an end of the text string. An example implementation of this claim element is described at Fig. 6, page 8, lines 2-4, page 9, lines 1-5, and page 9, lines 14-17.

D. Independent Claim 56

Independent claim 56 is directed to a method performed on a mobile communication device for receiving and displaying a command text string. The method includes the step of receiving a command text string being entered by a user. An example implementation of this claim element is described at Figs. 4 and 6, page 5, lines 6-16 and page 6, lines 9-10.

The method further includes the step of displaying a list of frequently used commands from the database as soon as the user begins entering the command text string,

for the user to select one of the commands from the list. An example implementation of this claim element is described at Fig. 6, page 8, lines 2-4, page 9, lines 1-5, and page 9, lines 14-17.

(VI) GROUND FOR REJECTION TO BE REVIEWED ON APPEAL

Claims 2, 5, 9-13, 37, 41-45 and 48-50 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 5,974,413 (hereinafter referred to as “the Beauregard Reference” or “Beauregard”) in view of non-patent reference Eide, *Valet: An Intelligent Unix Shell Interface*, Master’s Thesis, University of Utah, August 1995 (hereinafter referred to as “the Eide Reference” or “Eide”) and further in view of U.S. Patent No. 6,288,718 (hereinafter referred to as “the Laursen Reference” or “Laursen”). Claims 6-8 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Beauregard in view of Eide in view of Laursen and further in view of U.S. Patent No. 6,622,119. Claim 56 stands rejected under 35 U.S.C. § 103(a) as being unpatentable over Beauregard in view of U.S. Patent No. 7,216,292 (hereinafter referred to as “the Snapper Reference” or “Snapper”). These rejections are appealed.

(VII) ARGUMENT

A. **Independent claim 2**

Independent claim 2 generally relates to how an electronic device processes a user command for launching a software application. The device receives the command as it is being entered by the user and displays to the user a list of probable commands that match what the user entered so far.

Claim 2 is rejected over Beauregard in view of Eide and Laursen. This rejection is improper because claim 2 includes at least the following two limitations that are not disclosed by the references even in combination: (i) **while** receiving the command from the user, displaying a list of probable commands; and (ii) after the list is displayed, the user enters remaining characters of the command to narrow the list.

- i. The references fail to teach displaying a list
while command text is entered

Claim 2 recites the step of “while receiving the abbreviated textual command, performing the steps of... displaying a list of probable complete commands matching the currently received portion of the abbreviated textual command.” The rejection of claim 2 ignores the fact that this claim limitation requires the displaying step to be performed *while* receiving the abbreviated textual command. As explained in the specification: “[A] list of frequently used commands is displayed by the natural language search engine 32 as soon as the user begins entering text in the graphical dialog box 42. The user then has the option to continue typing or choose from the list of commands.” (Specification, page 9). In other words, the list of probable complete commands is displayed before the user enters any sort of indication that the abbreviated textual command has been completely entered

(e.g., by pressing enter or some other type of delimiter denoting an end of the text string.) Clearly, none of the cited references teach or suggest this claim element.

The Examiner has, on different occasions, pointed to both the Beauregard and Eide references as teaching this claim element.¹ The Applicants submit that neither of these references suggest displaying a list of probable commands “while” the command is being entered by the user. The Beauregard reference describes a user interface that allows a user to enter commands “in his everyday language” in order to control the operations of a computer. The cited portion of Beauregard explains that if the user designates a certain word for two or more operations, then a list is generated after the command is entered to allow the user to select the desired operation. (See, Beauregard, col. 42, lines 27-50). However, Beauregard repeatedly explains that no action, including the display of a list of possible operations, is performed until after the user has indicated the end of the command entry by entering a delimiter, such as by pressing the space bar twice. (See, Beauregard, col. 9, line 9; col. 15, line 49; col. 17, line 55; col. 38, line 25; col. 42, line 66; and col. 44, line 53).

The Eide reference, which is a master’s thesis relating to an intelligent UNIX interface, similarly states that his list is displayed only after the user presses the Tab key to denote end of command entry. Specifically, Eide explains: “[T]he user can type just: ‘Is sou’. At this point, before hitting the Return key, the user can have the shell complete the

¹ In the Office Action mailed on May 23, 2008, the Examiner indicated that this claim element is taught by Beauregard at col. 42, lines 27-50. (See, 5/23/08 Office Action, p. 8)(“With respect to Claim 2, Beauregard discloses: ... While receiving the abbreviated textual command performing the steps of: Searching ...; Displaying a list of probable complete commands ...Col. 42, lines 27-50”). In the latest Office Action mailed on December 4, 2008, the Examiner indicates at page 3 that this limitation is instead taught by Eide, but then again cites instead to Beauregard at page 9 of the Office Action. In any case, the claim limitation is not taught or suggested by either of these references.

partially entered directory name. The user can press the Tab key.” Eide continues explaining, in a footnote, that “If the prefix ‘sou’ had not been sufficient to uniquely identify exactly one file name, the shell ... would have completed as much of the user’s input possible or displayed a list of possible completions.” (See, Eide at page 37).

Accordingly, because none of the cited references teach or suggest the step of “while receiving the abbreviated textual command, performing the steps of... displaying a list of probable complete commands matching the currently received portion of the abbreviated textual command,” the Applicants submit that the rejection of claim 2 is improper and that the claim is patentable over the prior art.

- ii. The references fail to teach entering remaining characters of a command after the list is displayed

Claim 2 further recites the step of “if the user does not select a complete command from the list, then receiving the complete command as the abbreviated textual command from the user in the natural language search engine, by the user entering remaining characters of the entire abbreviated command to narrow the list of complete commands.” The Office Action asserts this is taught by Laursen. However, Laursen does not narrow a list of probable **commands**. (See, 12/4/08 Office Action, page 10.) Laursen instead narrows a list of **records** (col. 1, line 15) exemplified as names in an address list. (See, Laursen, col. 1, lines 15).

Laursen's teaching of successively narrowing a list of names in response to entered text would not motivate the skilled person to modify Beauregard to successively narrow a list of commands to arrive at claim 2. That is because the user would consider doing so to be uncalled-for in Beauregard's application and unlikely to succeed, for at least the

following four reasons: 1) In Laursen, the text being entered is a person's name, whereas in Beauregard, the text is a computer command. 2) In Laursen, the text is entered in response to a query (col. 5, line 67), whereas in Beauregard, the text is entered freestyle on the user's own volition (*e.g.*, entering "dial" command while typing an email in col. 44, line 42). 3) In Laursen, the type of text being entered is pre-defined (*e.g.*, name), whereas in Beauregard, the text is a command to perform any function the computer supports. 4) In Laursen, the user is entering an **unabbreviated** version of the text before the computer displays text suggestions. Whereas in Beauregard, the user's text command can be abbreviated (such as "msw" for Microsoft Word) or unabbreviated. Accordingly, for at least these four reasons, Laursen's teaching (of successively narrowing a list of **names**) would not suggest successively narrowing Beauregard's list of **commands** to arrive at claim 2.

The Applicants therefore submit that the rejection of claim 2 is also improper for this additional reason, and that the claim is patentable and in condition for allowance.

B. Independent claim 37

Independent claim 37 recites the method steps of (i) "before receiving the entire abbreviated textual command, comparing the received portion of the abbreviated textual command to the stored abbreviated commands to determine a probable subset of the complete commands"; and (ii) "displaying the probable subset of the complete commands to the user; and... if, instead of selecting a complete command, the user enters a further portion of the abbreviated textual command, then narrowing the probable subset based on said further portion." These claim elements are similar to the claim element of claim 2 discussed above, and the Office Action concludes that these elements of claim 37 are

obvious using the same rationale as for claim 2. (See, 12/4/08 Office Action, pages 12-13). The Applicants submit that the error in the Examiner's rationale for rejecting claim 37 is the same as explained above with reference to the rejection of claim 2, and that claim 37 is also patentable over the cited references and in condition for allowance.

C. Independent Claim 48

Independent claim 48 recites the method steps of “while receiving the text string, comparing a received portion of the text string to stored text commands to determine which of the stored text commands is a probable text command based on a portion of the probable text command matching the received text string; and initiating a software operation corresponding to the probable text command, *the comparing and initiating steps being performed without the user having entered a delimiter denoting an end of the text string.*” The Office Action rejects claim 48 on the same basis as claim 37. However, recognizing that the Beauregard and Eide references cited against claim 37 do not suggest “the comparing an initiating steps being performed without the user having entered a delimiter denoting an end of the text string,” the Examiner instead asserts that this claim element is made obvious by the teachings of the Laursen reference at col. 2, lines 1-24. (See, 12/4/09 Office Action, page 14). The Applicant respectfully disagrees.

The Laursen reference describes a user interface for sorting records in a database, such as the names in an electronic address book. The cited portions of Laursen describes a user interface that provides a list of records in a structured database, such as names from an address book. As the user enters alphabetical characters, a progressively reduced list of items that start with the entered characters is displayed. (See, Laursen, col. 2, lines 1-24). The Applicants submit that the skilled person would not apply these teaching from Laursen

to the user interfaces of Beauregard for at least the four reasons explained above with reference to the rejection of claim 2. Accordingly, claim 48 is also patentably distinct from the cited references and in condition for allowance.

The Applicants also note that in the "Response to Arguments" section of the December 4, 2008 Office Action, the Examiner concludes that the element of claim 48 relating to the delimiter is not supported in the specification and states that "the new matter rejection has been set forth below." However, the Office Action does not include a new matter rejection. For completeness, however, the Applicants have explained that this limitation is indeed supported in the application, by at least the following passages in conjunction with Fig. 6 (below):

"[T]he natural language search engine 32 will display a list of possible commands to the user in step 86. In step 88, the user may select from the list of possible commands, or alternatively may narrow the list by entering more text."
(application, p.8, lines 2-4)

and

"For example, the user may chose ... **"e_j'** to always represent the user command 'email jim.' In this manner, **only two key strokes** are required to invoke the email composer application 34, and select the addressee 'jim.' " (application, p.9, lines 1-5, emphasis added)

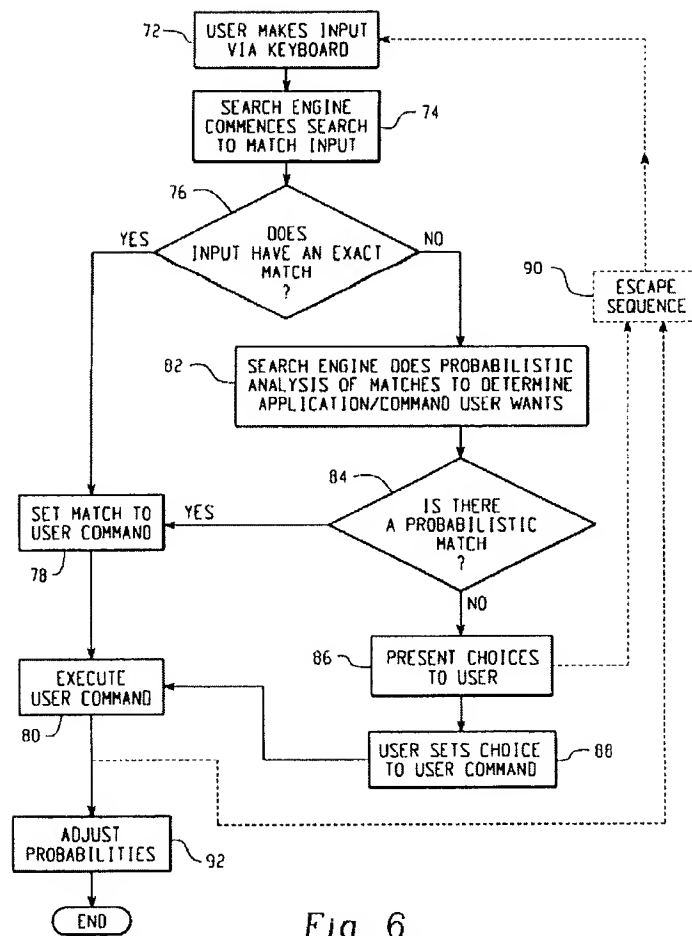


Fig. 6

D. Independent claim 56

Claim 56 recites the method step of “displaying a list of frequently used commands from the database **as soon as** the user begins entering the command text string, for the user to select one of the commands from the list.” The Office Action acknowledges that this limitation is not taught by the Beauregard reference, but concludes that it is made obvious by the Snapper reference. The Applicants respectfully disagree.

The Snapper reference describes a user interface for entering data into a form on a web page. The cited portions of Snapper explain that when the user begins entering text into a field of the online form, a drop-down list of possible choices appears that the user may select from. As the user begins typing, the entries in the list are reduced to those matching the already entered text. (See, Snapper, col. 10, lines 30-60). However, Snapper does not teach displaying probable **commands** while the user enters a command string; in fact, Snapper is irrelevant to entering commands. Snapper instead teaches displaying probable **personal information** while the user enters personal information. (See, e.g., Snapper, col. 7, lines 3-19).

Snapper's teaching (of displaying suggested names while the user enters his name) would not motivate the skilled person to modify Beauregard's display of probable **commands** to occur while the user enters an abbreviated command to arrive at claim 56. That is because the user would consider doing so to be uncalled-for in Beauregard's application and unlikely to succeed, for the following five reasons: 1) In Snapper, the text being entered is the user's personal information, whereas in Beauregard, the text is a computer command. 2) In Snapper, the text is entered in response to a website's query, whereas in Beauregard, the text is entered on the user's own volition. 3) In Snapper, the type of data being entered is pre-defined (for example, pre-defined to be a name in response a "NAME" query). Whereas in Beauregard, the text is a command to perform any function or application that the computer supports. 4) In Snapper, the user must start out entering an **unabbreviated** version of the text before the computer displays text suggestions. Whereas in Beauregard, the user's text command can be abbreviated (such as "msw" for Microsoft Word) or unabbreviated. 5) In Snapper, the browser can reasonably

assume that the user intends to respond to the website's query for "NAME" with the same answer that he previously gave for a "NAME" query. In contrast, such an assumption is irrelevant to Beauregard's application where the commands are entered free-form and not in response to a question. For these reasons, Snapper's teaching (of displaying probable **names** WHILE the user enters his name) would not suggest modifying Beauregard's display of **commands** to occur WHILE the user enters an abbreviated command to arrive at claim 56.

The Applicants therefore submit that the rejection of claim 56 is also improper, and that the claim is patentable and in condition for allowance.

E. Dependent Claims 5-13, 41-45 and 49-50

The remaining claims all depend from base claims that are explained above to be patentable over the prior art. The limitations that the dependent claims add to the base claims distinguish them further from the prior art.

(VIII) CLAIMS APPENDIX

A claims appendix containing a copy of the claims subject to this appeal is attached.


(IX) EVIDENCE APPENDIX

No evidence is being submitted pursuant to 37 C.F.R. 1.130, 1.131 or 1.132, nor is there any other evidence entered by the Examiner or relied on by the Applicant. An evidence appendix indicating "None" is attached.

(X) RELATED PROCEEDINGS APPENDIX

There are no proceedings related to this application. A related proceedings appendix indicating "None" is attached.

Respectfully submitted,
JONES DAY



Joseph M. Sauer
(Reg. No. 47,919)

Jones Day
North Point, 901 Lakeside Avenue
Cleveland, Ohio 44114
(216) 586-7506

CLAIMS APPENDIX

1 (canceled)

2 (previously presented): A method of launching a software application in a hand-held device, comprising:

receiving an abbreviated textual command in a natural language search engine, entered by a user of the hand-held device; and

while receiving the abbreviated textual command, performing the steps of:

searching a natural language database that stores a data set of abbreviated textual commands and associated application commands;

analyzing historical preferences to determine one or more probable complete commands matching a currently received portion of the abbreviated textual command;

displaying a list of probable complete commands matching the currently received portion of the abbreviated textual command;

if the user selects a complete command from the list, then setting the complete command as the abbreviated textual command, and executing the complete command; and

if the user does not select a complete command from the list, then receiving an entire abbreviated textual command from the user in the natural language search engine, by the user entering remaining characters of the entire abbreviated command to narrow the list of complete commands.

3-4 (canceled)

5 (previously presented): The method of claim 2, further comprising:

if the abbreviated textual command has an exact match in the data set, then setting the exact match as a user command;

if the abbreviated textual command does not have an exact match in the data set, then analyzing historical preferences to determine if the abbreviated textual command has a probable match in the data set;

if the abbreviated textual command has a probable match in the data set, then setting the probable match as the user command;

if the abbreviated textual command does not have a probable match in the data set, then presenting a list of possible commands, receiving a command choice, and setting the command choice as the user command; and

executing the user command.

6 (previously presented): The method of claim 2, wherein the step of analyzing historical preferences is performed using a set of probability factors that are generated based on historical preferences, where the abbreviated textual command has a probable match in the data set when a probability factor associated with the probable match is greater than a predetermined value.

7 (previously presented): The method of claim 6, wherein the predetermined value is defined by a user.

8 (previously presented): The method of claim 6, comprising the additional step of:

adjusting the set of probability factors each time the abbreviated textual command is entered into the hand-held device.

9 (previously presented): The method of claim 2, wherein:

the abbreviated textual command has a first component and a second component, wherein the first component represents a desired application command, and the second component represents a desired application tag; and

the natural language database stores a data set of abbreviated textual commands and associated application commands and tags.

10 (previously presented): The method of claim 2, wherein the abbreviated textual command is entered into a graphical dialog box.

11 (previously presented): The method of claim 2, wherein the natural language search engine can receive the abbreviated textual command while any of the software applications are executing.

12 (previously presented): The method of claim 5, wherein the list of possible commands presented if the abbreviated textual command does not have a probable match in the data set includes a set of recently executed application commands.

13 (previously presented): The method of claim 5, wherein the list of possible commands presented if the abbreviated textual command does not have a probable match in the data set includes a set of generic application commands that the natural language search engine is capable of executing.

14-36 (canceled)

37 (previously presented): A method comprising:

storing a data set of abbreviated textual commands and corresponding complete commands;

receiving a portion of an abbreviated textual command being entered by a user;
and

before receiving the entire abbreviated textual command, comparing the received portion of the abbreviated textual command to the stored abbreviated commands to determine a probable subset of the complete commands;

displaying the probable subset of the complete commands to the user; and

if the user selects one of the complete commands, then executing the selected complete command; and

if, instead of selecting a complete command, the user enters a further portion of the abbreviated textual command, then narrowing the probable subset based on said further portion.

38-40 (canceled)

41 (previously presented): The method of claim 37 further comprising:

when the probable subset consists of only one complete command, executing that one complete command.

42 (previously presented): The method of claim 37 wherein the storing step includes a user assigning which complete commands should correspond in the future to which abbreviated textual commands.

43 (previously presented): The method of claim 37 wherein the storing step includes generating the data set based on which abbreviated textual commands a user has historically used for choosing each complete command.

44 (previously presented): The method of claim 37 wherein the comparing step includes:
if the data set indicates that the user has chosen to execute a particular complete command more than a predetermined percentage of the time less than 100% after having entered an abbreviated textual command matching the currently received portion of text, then narrowing the subset to that command.

45 (previously presented): The method of claim 44 wherein the predetermined percentage is 50%.

46-47 (canceled)

48 (previously presented): A method comprising:

receiving a text string being entered by a user;

while receiving the text string, comparing a received portion of the text string to stored text commands to determine which of the stored text commands is a probable text command based on a portion of the probable text command matching the received text string; and

initiating a software operation corresponding to the probable text command;

the comparing and initiating steps being performed without the user having entered a delimiter denoting an end of the text string.

49 (previously presented): The method of claim 48 wherein said portion of the probable text command is not the entire text command.

50 (previously presented): The method of claim 48 wherein the comparing step includes:

identifying a plurality of the stored text commands that have portions matching the received text string; and

determining which one of the plurality is the probable text command based on historical preferences.

51-55 (canceled)

56 (previously presented): A method performed by a mobile communication device, comprising:

receiving a command text string being entered by a user; and

displaying a list of frequently used commands from the database as soon as the user begins entering the command text string, for the user to select one of the commands from the list.

EVIDENCE APPENDIX

None

RELATED PROCEEDINGS APPENDIX

None